A Corpus and Semantic Parser for Multilingual Natural Language Querying of OpenStreetMap

Carolin Haas
Computational Linguistics
Heidelberg University
69120 Heidelberg, Germany
haas1@cl.uni-heidelberg.de

Stefan Riezler
Computational Linguistics & IWR
Heidelberg University
69120 Heidelberg, Germany
riezler@cl.uni-heidelberg.de

Abstract

We present a corpus of 2,380 natural language queries paired with machine readable formulae that can be executed against world wide geographic data of the OpenStreetMap (OSM) database. We use the corpus to learn an accurate semantic parser that builds the basis of a natural language interface to OSM. Furthermore, we use response-based learning on parser feedback to adapt a statistical machine translation system for multilingual database access to OSM. Our framework allows to map fuzzy natural language expressions such as "nearby", "north of", or "in walking distance" to spatial polygons on an interactive map. Furthermore, it combines syntactic complexity and compositionality with a reasonable lexical variability of queries, making it an interesting new publicly available dataset for research on semantic parsing.

1 Introduction

OpenStreetMap[1] (OSM) is a community-built database of geographic data, containing user-contributed local and up-to-date information about landmarks all over the world. Currently, the database contains over 3 billion data objects and is continuously growing with contributions from over 2 million registered users. While the main API is optimized for editing map data, there exists an API that allow to filter map data based on search criteria such as location, type of objects, or features with which objects are tagged. However, issuing a query that is executable against the OSM database still requires detailed knowledge of database internals, something that cannot be expected from a layman user.

The goal of our work is the development of an interface to OSM that lets a user ask a question in natural language, which is then parsed into a database query that is executable against a web-based filtering tool and returns OSM data on an interactive map. For example, we want a user without detailed knowledge of OSM to be able to ask questions that embrace the "fuzziness" of natural language, for example, "*What are the locations, names and telephone numbers of hotels in Paris with wheelchair access that are close to the station Gare du Nord?*". To find such information one would have to issue a query that requires detailed knowledge of the database and the query language: "*area[name='Paris']→.a;node(area.a)[name='Gare du Nord']→.b;node(around.b:1000)[tourism='hotel'][wheelchair='yes'];out;*". Additionally, we present an adaptation of a statistical machine translation (SMT) system for multilingual access to OSM by response-based learning from parser executability of translated queries.

As a starting point for our natural language interface we built a corpus of 2,380 natural language queries paired with machine readable language (MRL) formulae that we used to extract a semantic parser. We chose to manually creating a corpus of MRLs from which structure and weights of a semantic parser can be learned for three reasons:

Online availability: We want to be able to present the OSM community with a set of sample questions that can be executed and whose database

[1] http://www.openstreetmap.org

query representation can be inspected. We hope it will be inspiring and helpful for OSM users and developers to see how complex geographical facts can be issued as simple natural language queries that are parsed into executable filters on OSM objects.

Accuracy: For an online natural language interface, high accuracy of semantic parsing is crucial. So far, semi-automatic methods of constructing semantic parsers without bootstrapping from MRLs could not reach the accuracy of semantic parsers extracted from manually created MRLs (Wang et al., 2015). The semantic parser we extracted by simple monolingual machine translation (Andreas et al., 2013) achieves a F1 score of 77.3% for answer retrieval on our data.

Complexity: Our corpus adds a new and complex domain for research on semantic parsing: Compared to existing corpora such as GEOQUERY (Wong and Mooney, 2006) or FREE917 (Cai and Yates, 2013), our corpus combines the compositionality and syntactic complexity of the former corpus with a lexical variation that constitutes a healthy middle ground between closed and open domains.

Our contributions in this paper are threefold: First, we introduce OSM as a new knowledge base that has not, to the best of our knowledge, been used for question answering, and offer a new corpus to the research community. Second, we show that a parser read off the corpus achieves promising parsing accuracy and can be used to adapt SMT to multilingual database access. Third, our work builds the basis of a natural language interface to OSM that will be enabling for interesting directions of future research, e.g., response-based learning to improve semantic parsing and multilingual database access.

2 Related Work

The common approach to semantic parsing is a manual annotation of a corpus with natural language utterances and machine readable formulae which are then used to learn the structure and weights of a semantic parser. Corpora that have been used for training and testing a number of semantic parsers are GEOQUERY (Zelle and Mooney, 1996; Kate et al., 2005) and FREE917 (Cai and Yates, 2013). While GEOQUERY queries are restricted to the closed domain of US geography, the structural complexity of the questions is higher than for FREE917, which focuses on open domain queries. Seminal work on building semantic parsers from the GEOQUERY meaning representations are Zettlemoyer and Collins (2005) or Wong and Mooney (2006). Later approaches try to learn semantic parsers from question-answer pairs only, for example, Liang et al. (2009) for GEOQUERY, or Kwiatkowski et al. (2013) or Berant et al. (2013) for FREE917. Newer research attempts to close the gap between lexical variability and structural complexity (Vlachos and Clark, 2014; Artzi et al., 2015; Pasupat and Liang, 2015), however, answer retrieval accuracy is low if semantic parsers cannot be bootstrapped from a corpus of queries and MRLs (Wang et al., 2015; Pasupat and Liang, 2015).

# users	2,389,374
# objects	3,464,399,738
# nodes	3,139,787,926
# ways	320,775,580
# relations	3,836,232
# tags	1,259,132,137
# distinct tags	76,204,309
# distinct keys	57,159

Table 1: Statistics of OSM as of December 14th, 2015

Our approach treats semantic parsing as a monolingual machine translation problem in which natural language is translated into the machine readable language. This approach is convenient because one can make use of the efficient and robust decoders that are freely available for SMT. Despite the simplicity of the approach, Andreas et al. (2013) have shown that highly accurate semantic parsers can be trained from annotated data.

OSM has previously been used by Boye et al. (2014) for pedestrian routing using a dialogue system, however, no details on semantic parsing and no resource are provided.

Our SMT tuning experiment builds on the work of Riezler et al. (2014) and Haas and Riezler (2015) who applied response-based learning for SMT to the GEOQUERY and FREE917 domains, respectively.

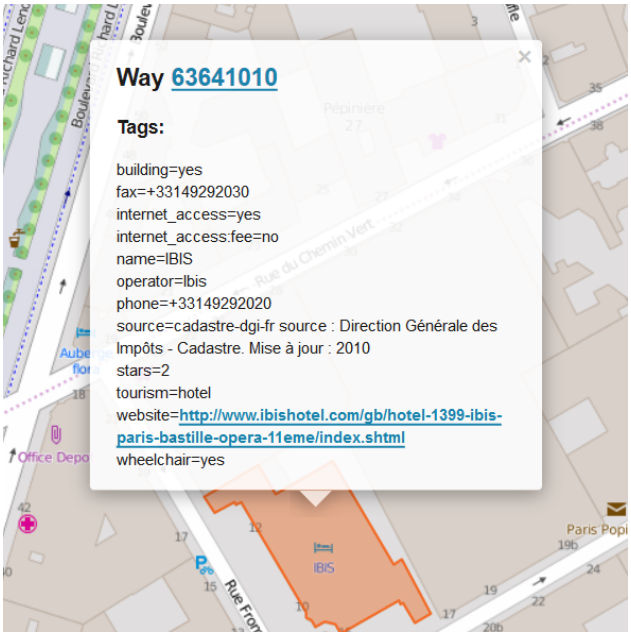

Figure 1: Illustration of a possible entity (here: way) on the Overpass turbo website for the query "*area[name='Paris']→.a;node(area.a)[name='Gare du Nord']→.b;node(around.b:1000[tourism='hotel'][wheelchair='yes'] ;out;*". The way consists of several nodes that together span the outline of the relevant building. Clicking on the way gives all the key-value pairs registered for this way in the database.

3 OpenStreetMap

OpenStreetMap is a freely available map of the world, annotated by volunteers and editable by anyone. Entered GPS points, referred to as *nodes*, constitute the basis of the database and currently amount to over 3 billion examples (see Figure 1 for more statistics on the OSM database). Nodes can be given tags which are key-value pairs, such as "*amenity=restaurant*", "*highway=living_street*" or "*abandoned:tourism=theme_park*". In total there are over 76 million distinct tags which are based on 57,159 unique keys. Nodes may be grouped together to form *ways*. Ways can be given their own set of tags and are for example used to display roads or building outlines. Both nodes and ways may be joined to be part of a *relation* which is used to model the interdependence of several objects. A relation can for example be employed to delineate bus lines or to define administrative boundaries. As relations can also be part of other relations, one can even express hierarchical structures.

The Overpass API[2] can be used to query the database made up of the aforementioned nodes, ways and relations. It can efficiently extract the correct subset of database objects that satisfy the entered constraints. The following constraints are most relevant for our corpus creation later on:

- The simplest constraints require the database objects to have certain keys or key-value pairs. For example to search for hotels, one would use "*node[tourism='hotel'];out;*".
- Overpass can find ways and relations that form a filled polygon. Based on this, Overpass is able to search for database objects in only the specified *area*, such as a town or country. To search for hotels in Paris, the corresponding Overpass would be: "*area[name='Paris']→.a; node(area.a)[tourism='hotel'];out;*".
- The operator called *around* allows the user to search for a database object in a certain radius around another object. For hotels in a radius of 1,000 metres around Gare du Nord in Paris, the Overpass query would be: "*area[name='Paris']→.a; node(area.a)[name='Gare du Nord']→.b; node(around.b:1000)[tourism='hotel'];out;*".

Further tools related to OSM are Overpass turbo[3], a web interface that allows users to run Overpass queries, and the Overpass turbo Query Wizard, which supports querying by predefined human readable shorthands to executable Overpass queries. A screenshot of the map and an example database entry for an Overpass query is shown in Figure 1.

4 Query Creation

Since even the Overpass Query Wizard requires users to be familiar with the tag set of OSM key-value pairs, it unusable for users with only casual or no knowledge of OSM's internal structure. Nonetheless, we could use parts of the user query log to formulate natural language questions. For example users would enter the query "*(node["abandoned:tourism"="theme_park"]; way["abandoned:tourism"="theme_park"]; rela-*

[2] http://wiki.openstreetmap.org/wiki/Overpass_API

[3] http://overpass-turbo.eu/

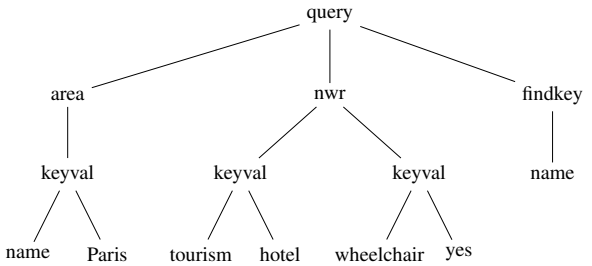

Figure 2: The question "*Which hotels in Paris have wheelchair access?*" with MRL "*query(area(keyval(name,'Paris')), nwr(keyval(tourism,'hotel'),keyval(wheelchair,'yes')), findkey(name))*" can be presented as a tree. A preorder traversal gives: "*query@3 area@1 keyval@2 name@0 Paris@s nwr@2 keyval@2 tourism@0 hotel@s keyval@2 wheelchair@0 yes@s findkey@1 name@0*"

tion["abandoned:tourism"="theme_park"];);out;" which we then extended to "When did the abandoned theme parks close?" and added the corresponding MRL "*query(nwr(keyval('abandoned:tourism','theme_park')),qtype(findkey('end_date')))*". Additionally we often had to provide a reference point for the queries, as users usually searched their current map cut-out and while all abandoned theme parks in the world is a short list, all restaurants for example would be too many to realistically handle. Thus we fell back to 3 chosen cities in those cases, Heidelberg, Paris and Edinburgh. In sum, most of the queries are based on OSM user queries that were issued to the Overpass turbo Query Wizard and shared among users; others were written by the first author with an utilization of the underlying Overpass API in mind.

5 Machine Readable Language (MRL)

Our corpus creation process was guided by the goal to pair a diverse range of questions with machine readable language (MRL) formulae. These should include the most important OSM tags so that the parser is able to learn a mapping between these tags and the different corresponding natural language expressions.

To answer concise questions without including superfluous details, the MRL needs to be able to extract more specific information instead of a list of all database objects as in the underlying Overpass result. The MRL thus wraps around Overpass and contains additional indicators about what information should be returned from a database object, for example just its GPS coordinates, or a website address, or the number of returned objects.

Given the above consideration, we define our MRL as a variable free language that focuses on practicality and speed, akin in style to the GEOQUERY MRL language. It is unambiguously defined via a context-free grammar (CFG) so that one can always ascertain whether or not a formula is valid. While the written form of the MRL is a bracket structure, this structure can easily be encoded as a tree by taking a pre-order traversal which makes it easy and efficient to work with. An example CFG tree for a MRL is given in Figure 2. In the following, we list the operators of our MRL.

Query Operator. A single database query is encoded in the operator **query()** which will hold the Overpass query as well as further specifications about what kind of answer should be retrieved. A few operators are directly derived from Overpass, merely re-written as a tree structure. As such OSM key-value pairs are encoded using the operator **keyval()** which takes 2 arguments, the first being the *key* and the second the *value*. The area operator from Overpass directly translates to the operator **area()**. Nodes, ways and relations are grouped together under the **nwr()** operator which will supply the union of the query run with the 3 types in turn. This is necessary because often buildings, e.g. schools, may be represented as any of the 3 types depending on how specific the annotator wanted to be. Both *area()* and *nwr()* then take one ore more **keyval()** arguments. If *area()* and *nwr()* appear as siblings in the tree (for an example see Figure 2), then only the objects that lie within *area()* will be searched to determine if they fulfil the *nwr()* constraints.

Meta Operators. In order to add specificity beyond the lists returned by Overpass, each MRL formula needs one or more of the following meta parameters to be valid; the meta parameters in turn are held by the operator **qtype()**. The operator **latlong()** retrieves the geographical coordinates of the database objects. For a node this is simply its recorded GPS point. In the case of ways and relations the centre of the associated nodes is calcu-

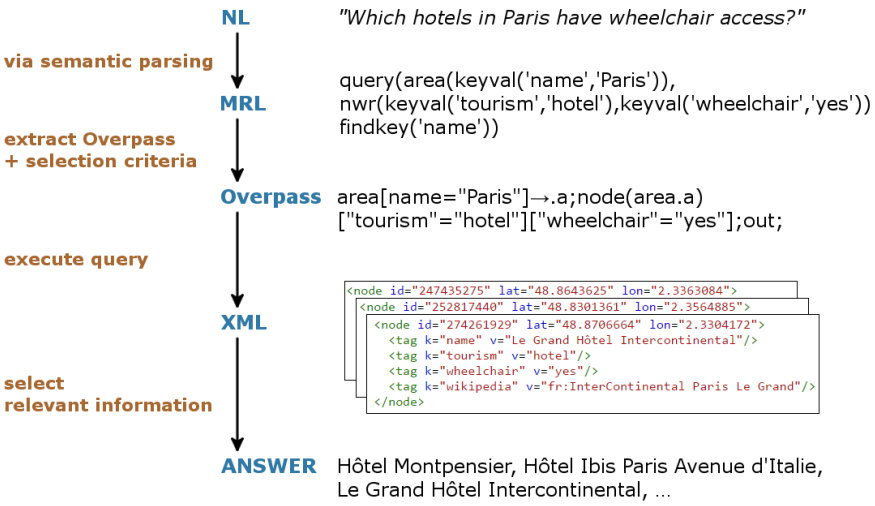

Figure 3: Mapping of natural language question to MRL (via semantic parsing) and to structured database query, returning a set of database objects (via Overpass API) which may be represented as XML documents, from which the correct answer is retrieved.

lated. **findkey()** searches for a specific key (such as *name* or *website*) in the database objects of the retrieved set and returns its value. **count()** simply counts the number of elements in the retrieved set. **least()** checks for the existence of at least *x* elements in the returned set, whereas *x* is defined in a second meta function **topx()** which returns the top *x* elements of a set. *topx()* may also be used in conjunction with *latlong()* and *findkey()*.

Additionally the user can ask for the distance between two points of interests. This operator, **dist()** needs to be supplied with two separate *query()* operators using the *latlong()* meta operator, and can return the value in either kilometres or miles (*unit(mi/km)*). In conjunction with that the user can inquire if the point of interest is still within walking distance (*for("walk")*), or if a car is recommended (*for("car")*).

Fuzzy Language Operators. Fuzzy terms such as "nearby", "within walking distance" and "closest" can be modelled by making use of the *around* operator from Overpass. **around()** searches for points of interest (supplied via **search()**) in the vicinity of another (supplied using **center()**). If only the *x* closest points are to be returned, *topx()* can be added. The radius is defined via *dist()*. This information can occur either explicitly (*"No further than 200m away"*), or implicitly (*"Give me a cinema with a car park close by"* implies that the car park should be in walking distance). For the implicit case 4 options are available: walking distance, within town distance, out of town distance, and day trip distance. Choosing the appropriate distance in the implicit case is of particular difficulty because often a term such as "*close by*" implies a different distance depending on the surrounding context. For example, "*a close by airport*" may imply day trip distance, while "*a close by restaurant*" at most implies a just out of town distance (see also Minock and Mollevik (2013)).

Another set of fuzzy terms are the cardinal directions, either within an *area()* operator ("*Where are hotels in the north of Paris?*"), or beyond an *nwr()* operator ("*Where are hotels north of Gare du Nord in Paris?*"). The correct operator, **north()**, **east()**, **south()** or **west()**, follows after *query()*, if present.

Further Operators. Some further operators were needed to model the MRL formula for complex questions. **and()** is used when the user asks for two different nuggets of information (*"Where is the closest bakery and the closest butcher?"* or *"Give me the website and name of ..."*). **or()** is used to create unions, as for example, needed in a sentence such as *"Give me the closest bar or restaurant."* "*" can be used as a wild card in a value position, e.g. *['historic'=*]* will returned any historic objects, be it a castle, a monument or something else. **nodup()** returns a set with no duplicates. This is, for example, needed in *"Which cuisines are there?"*.

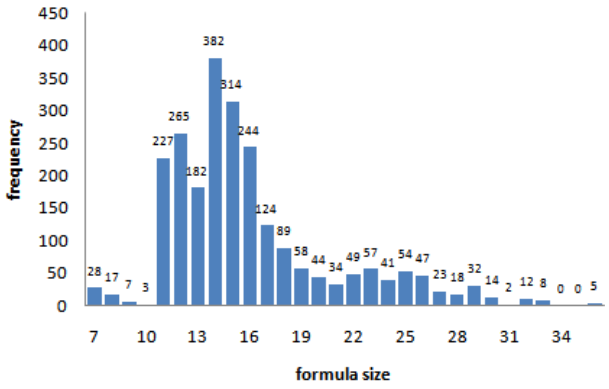

Figure 4: Histogram of the gold formulae sizes

	NLMAPS	GEOQUERY	FREE917
# sent.	2,380	880	917
tokens	25,906	6,660	6,785
types	1002	296	2,038
avg. sent. length.	10.88	7.57	7.4
avg. types per sent.	0.42	0.34	2.22
avg. singleton per sent.	0.1	0.1	1.52
avg. NT per sent.	21	16	16
FRES	82.18	86.61	83.77

Figure 5: Corpus statistics of the corpora NLMAPS, GEOQUERY, FREE917. NT stands for non-terminal and FRES is the Flesch Reading Ease Score (Flesch, 1974).

The complete pipeline process leading from natural language question to an appropriate answer can be seen in Figure 3. The natural language question is first translated by the semantic parser into a MRL, from which the Overpass query is deterministically extracted, while keeping a list of the relevant further indicators. These indicators then operate on the database objects returned by the Overpass query to form the answer.

6 Corpus Statistics

Our corpus, called NLMAPS, is more than twice as large as GEOQUERY or FREE917. In the following, we present a comparison of the lexical and syntactic complexity of the three corpora. All statistics reported in Figure 5 are normalized by the number of sentences.

Lexically, NLMAPS is more diverse than GEOQUERY, as can be seen by the average number of types, but less so compared to FREE917 due to the fact that the OSM database is still a somewhat more closed domain compared to Freebase. Syntactically however, NLMAPS is with 3 more words on average per sentence more complex than GEOQUERY and FREE917, which have nearly identical sentence length. As a further test, we ran the Stanford Parser (Klein and Manning, 2003) on the queries to generate syntactic parse trees. We then counted the number of non-terminals required to produce the parse tree. This result reaffirms what the simpler sentence length already reported: the language in NLMAPS is more complex than in the other two corpora, which have identical complexity.

In Figure 4 we report the number of operators and values needed to construct the different gold formulae. While there are a few questions that need a formula shorter than 10, the vast majority needs a length of around 15, followed by a long tail of sizes with decreasing frequency of up to 36. The fact that many of the gold formulae are in fact longer than the average sentence length shows that the questions are far from trivial and require elaborate database queries to be answered. The last measure we report is the Flesch Reading Ease Score (Flesch, 1974) which is usually used to asses the text difficulty for readers. In this score, a lower number indicates a harder text. NLMAPS receives the lowest number, indicating it as the most complex corpus.

Overall we can infer that NLMAPS provides a good balance between lexical diversity as well as stability for a machine learning algorithm to learn. With regards to syntactic complexity, NLMAPS easily supersedes the other two corpora. We conclude this section with a few example sentences:

```
What are the websites and
names of the museums or art
centers in walking distance of
the Eiffel Tower?

What are the opening times of
the Sainsbury's Local closest
to the Edinburgh Waverley in
Edinburgh?

What are the peaks in the
north of Languedoc-Roussillon
called and how high are they?
```

7 Semantic Parsing

We treat semantic parsing as an SMT problem, using our own implementation of the framework in-

		Precision	Recall	F1
1	+intersect +stem +cdec	84.69	62.42	71.87
2	+intersect +stem +cdec +sparse	84.40	65.8	73.95[1]
3	+intersect +stem +cdec +pass +cfg	**89.45**	65.19	75.41[1]
4	+intersect +stem +cdec +sparse +pass +cfg	89.04	**68.3**	**77.3**[1,2,3]

Table 2: Semantic Parsing results on NLMAPS (split 1500/880 for train/test set) using different settings. Tuning was carried out on the training set. In the case of MERT tuning, the results are averaged over 3 runs due to the randomness MERT introduces. Best results are indicated in **bold face**. Statistical significance in terms of F1 of system differences at $p < 0.05$ are indicated by experiment number in superscript.

troduced by Andreas et al. (2013) who have shown that this approach achieves state-of-the-art performance on GEOQUERY. In this framework it is crucial that the MRL can be represented as a tree. A pre-order tree traversal can give a unique string in which each node is a word (i.e. surrounded by white space). Once the MRL has been converted into such a structure (for an example see Figure 2), a word aligner, here GIZA++ (Och and Ney, 2003), can be used to generate word-to-word alignments in both translation directions which can then be combined with various heuristics (Koehn, 2010, Chapter 4.5.3). From the next step onwards we use the freely available SMT framework CDEC (Dyer et al., 2010). After building a language model for the target (MRL) side, SCFG grammars for hierarchical phrases for tuning and testing were extracted. Experiments in n-gram order showed that 5-gram models are sufficient for language modelling.

At test time, a critical issue is the fact that monolingual SMT does not ensure that the translations are valid MRL formulae. Thus a k-best list (sorted from most probable to least) is generated which needs to be traversed until a valid formula is found. Experimentation with the k-best list size showed that 100 is a good trade-off between speed and performance. A bigger size of k might enable us to find a valid (and correct) formula further down the k-best, however, we verified experimentally that in most cases no option in the extended k-best list contained a valid formula either.

Once a valid formula is found, it is executed against a database and the resulting answer is compared to the answer the gold formula provides. Only exact matches are considered correct. We report *Precision*, *Recall* and *F1-score* to evaluate the semantic parser.[4]

Table 2 shows experimental results for different settings of semantic parsing on NLMAPS. Statistical significance of system differences in terms of F1 was assessed by an Approximate Randomization test (Noreen, 1989). For the word-alignment step, we found that the choice of the strategy for combining word alignments from both translation directions is crucial to the semantic parser's performance. The *intersect* strategy performs significantly better than any other, suggesting that high precision alignments are very important when using a monolingual SMT approach for semantic parsing. Further, stemming the words on the NL side is also always significantly better than not doing so. In a next step we compare the use of dense features (Dyer et al., 2010) in conjunction with the tuning algorithm MERT (Och, 2003) and additional sparse features (Simianer et al., 2012). As MERT cannot handle such a large amount of features, we paired the sparse features with the tuning algorithm MIRA (Crammer and Singer, 2003). Because MERT suffers from optimizer instability (Clark et al., 2011) due to random initialization, the experiments 1 & 3 in Table 2 report the average result based on 3 different MERT runs. Another variation we tested is the use of the NLMAPS CFG to check whether or not a translation is a valid MRL, instead of a quicker check that only ensures that a translation can be parsed as a tree. This variation is indicated with "+*cfg*". Lastly, while the system was able to directly learn the mapping of named entities previously seen dur-

[4] *Recall* is defined as the percentage of correct answers out of all examples, *Precision* the percentage of correct answers out of all examples with an answer, and *F1-score* the harmonic mean between the two aforementioned.

ing training, this is not possible for new named entities. These unknown named entities will automatically be passed through by the SMT system but they will be missing the marker of where in the tree they belong. As named entities always have to be in a value position in the corresponding MRL and values are always leave nodes, this can easily be rectified by appending the marker for a leave node to passed through words ("+*pass*"). Of course, when stemming is used, one has to also keep track of the unstemmed form. The decision of which route to go with named entities is left up to the SMT system.

Overall, the modules tested in Table 2 add up to a total F1 score of 77.3%. Given the complexity of our corpus and the simplicity of the semantic parser, this is a promising result.

8 Multilingual Parsing

Riezler et al. (2014) and Haas and Riezler (2015) have shown how to use semantic parsers for GEOQUERY and FREE917, respectively, to adapt an SMT system for multilingual database access. In this section we show that our semantic parser can be used for response-based learning of an SMT system to allow multilingual natural language queries to OSM, here German. To this end, a SMT system first translates the question from German to English and then the translation is passed to the semantic parser to answer the question. The SMT system uses the feedback from the parser, i.e. the knowledge whether or not the question could be parsed to the correct answer, to improve translations.

More formally, assume an SMT system with a joint feature representation $\phi(x, y)$ for input sentences x and output translations $y \in Y(x)$, that uses a linear scoring function to predict the most likely translation $\hat{y} = \arg\max_{y \in Y(x)} w^\top \phi(x, y)$. We define a cost function $\mathrm{c}(y^{(i)}, y) = (1 - \mathrm{BLEU}(y^{(i)}, y))$ based on sentence-wise BLEU (Nakov et al., 2012) and a binary feedback function $\mathrm{e}(y) \in \{1, 0\}$. The binary function evaluates to 1 if and only if a natural language's semantic parse receives the same answer as the corresponding gold parse. Training is performed by moving w closer to a *hope* translation y^+ while pushing it away from a *fear* translation y^-. Both y^+ and y^- have a high model score. y^+ incorporates the *hope* for a best translation to have a low cost and positive feedback. y^- is *feared* due to a high cost and positive feedback, thus we define:

$$y^+ = \underset{y \in Y(x^{(i)}):\mathrm{e}(y)=1}{\arg\max} \Big(\mathrm{s}(x^{(i)}, y; w) - \mathrm{c}(y^{(i)}, y) \Big),$$

$$y^- = \underset{y \in Y(x^{(i)}):\mathrm{e}(y)=0}{\arg\max} \Big(\mathrm{s}(x^{(i)}, y; w) + \mathrm{c}(y^{(i)}, y) \Big).$$

The algorithm, called REBOL (Riezler et al., 2014), proceeds by iterating over the training data, predicting the top translation $\hat{y}$, and receiving feedback for this translation from a semantic parser. If the feedback is positive, $\hat{y}$ is set equal to y^+, otherwise to y^-. The algorithm then searches the k-best list for the missing y^- or y^+, respectively, and performs an update that adds the feature vector of y^+ onto w, and subtracts the feature vector of y^-.

For our experiment, the NLMAPS questions were translated by the first author into German. As parser we chose to use number 3 (+intersect +stem +cdec +pass +cfg) from Table 2, deciding against the use of sparse features due speed reasons. The CDEC (Dyer et al., 2010) decoder was used for machine translation from German to English. Here we employ its standard features plus additional sparse features[5] and the COMMON CRAWL[6] (Smith et al., 2013) corpus to built the baseline SMT system.

REBOL is compared to a baseline system without discriminative training (CDEC) and to a stochastic (sub)gradient descent variant of RAMPION (Gimpel and Smith, 2012). Both baseline systems do not make use of the feedback from the semantic parser. While both REBOL and RAMPION assume the availability of both a reference translation and a gold parse, response-based learning can also succeed without any access to reference translations or even to gold standard parses. Riezler et al. (2014) introduced an algorithm, called EXEC, that only relies on task-based feedback and omits the cost function based on sentence-wise BLEU. Collecting real world data for this algorithm is realistic for an online interface to OSM since it only requires a user to pose a question and then indicate if it was answered to their satisfaction.

[5] https://github.com/pks/cdec-dtrain

[6] http://www.statmt.org/wmt13/training-parallel-commoncrawl.tgz

method	P	R	F1	BLEU
1CDEC	67.8	24.89	36.41	38.3
2EXEC	75.2	31.36	44.27[1]	40.85[1]
3RAMPION	78.21	38.75	51.82[1,2]	51.82[1,2]
4REBOL	80.76	41.02	**54.41**[1,2,3]	**51.88**[1,2]

Table 3: SMT results on NLMAPS, reporting Precision (P), Recall (R) and their harmonic mean (F1). Best results are indicated in **bold face**. Statistical significance of result differences at $p < 0.05$ are indicated by algorithm number in superscript.

While we do report BLEU (Papineni et al., 2002), the primary goal in our work is to achieve highest possible F1 score. This is vital because our ultimate aim is to give users asking German questions the correct answer, whereas the English translation from which BLEU is be calculated is only an intermediate result that is irrelevant for the task goal.

To test significance of F1 and BLEU, we again use Approximate Randomization. Before training, we split of 200 sentences from the training set to use as held out data (dev set). RAMPION, REBOL and EXEC ran for 50 epochs and then the dev set was used to pinpoint the best epoch for each algorithm. In the case of RAMPION, the best epoch equalled the epoch in which the dev set achieved the highest BLEU score (epoch 20). For REBOL and EXEC, on the other hand, this decision was made by the highest F1 score on the dev set (epoch 40 and 6 respectively). The learning rate for both algorithms was set on a per feature basis using Adadelta (Zeiler, 2012).

As shown in Table 3, REBOL can significantly improve in terms of F1 and BLEU over the CDEC baseline. It is also significantly better than RAMPION in terms of F1 while being able to keep up in BLEU. Should reference translations not be available, EXEC shows that it can still significantly outperform the CDEC baseline, it however cannot keep up with REBOL or RAMPION which have a more detailed supervision signal available to them.

9 Conclusion

We presented an approach to query the OSM database for complex geographical facts via natural language questions. The key technology is a semantic parser that is trained in supervised fashion from a large set of questions annotated with executable MRLs. Our corpus is larger than previous annotated question-answer corpora, while including a wide variety of challenging questions.

Terms such as "*nearby*", "*in the south of*", "*within x miles*" are particularly well-suited for a natural language query interface that allows to map the fuzziness of natural language to flexible spatial polygons. Our corpus is publicly available[7] and a website where users can query OSM using natural language is under development[8]. This in turn will give us new and more realistic data which we can use to extend the corpus and to improve the semantic parser.

An online version of our natural language interface to OSM will be enabling for various interesting directions of future research: Besides the possibility to gain new and more realistic data which we can use to extend the corpus, the semantic parser can be improved itself by response-based learning, where supervision signals can be extracted from the executed parses of new user queries against the database (Kwiatowski et al. (2013), Berant et al. (2013), Goldwasser and Roth (2013), *inter alia*). In a similar way, multilingual database access can be enhanced by adapting an SMT system by response-based learning, using executability of a parse of a translated query as supervision signal (Riezler et al., 2014). Both cases of response-based learning only require a user who issues a query and gives feedback on whether the proposed OSM object was the intended answer. Such an interactive scenario enables further research on alternative algorithms for learning from partial feedback (Szepesvári, 2009; Bubeck and Cesa-Bianchi, 2012).

Acknowledgments

We would like to thank the OSM developers Roland Olbricht and Martin Raifer for their support and for contributing a dataset of shared user queries. The research reported in this paper was supported in part by DFG grant RI-2221/2-1 "Grounding Statistical Machine Translation in Perception and Action".

[7] www.cl.uni-heidelberg.de/nlmaps

[8] nlmaps.cl.uni-heidelberg.de

References

Jacob Andreas, Andreas Vlachos, and Stephen Clark. 2013. Semantic parsing as machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, Sofia, Bulgaria.

Yoav Artzi, Kenton Lee, and Luke Zettlemoyer. 2015. Broad-coverage CCG semantic parsing with AMR. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, Lisbon, Portugal.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, WA.

Johan Boye, Morgan Fredriksson, Jana Gtze, and Jrgen Knigsmann. 2014. A demonstration of a natural-language pedestrian routing system. In *Proceedings of the 5th international workshop on spoken dialogue systems (IWSDS)*, Napa, USA.

Sébastian Bubeck and Nicolò Cesa-Bianchi. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extenstion. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, Sofia, Bulgaria.

Jonathan Clark, Chris Dyer, Alon Lavie, and Noah Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, Portland, OR.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, Uppsala, Sweden.

Rudolf Flesch. 1974. *The Art of Readable Writing: With the Flesch Readability Formula*. Harper & Row.

Kevin Gimpel and Noah A. Smith. 2012. Structured ramp loss minimization for machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, Stroudsburg, PA.

Dan Goldwasser and Dan Roth. 2013. Learning from natural instructions. *Machine Learning*, 94(2):205–232.

Carolin Haas and Stefan Riezler. 2015. Response-based learning for machine translation of open-domain database queries. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Denver, CO.

Rohit J. Kate, Yuk Wah Wong, and Raymond J. Mooney. 2005. Learning to transform natural to formal languages. In *Proceedings of AAAI*, Pittsburgh, PA.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 423–430.

Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, WA.

Tom Kwiatowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, WA.

Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning dependency-based compositional semantics. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, Portland, OR.

Michael Minock and Johan Mollevik. 2013. Context-dependent 'near' and 'far' in spatial databases via supervaluation. *Data Knowl. Eng.*, 86:295–305.

Preslav Nakov, Francisco Guzmán, and Stephan Vogel. 2012. Optimizing for sentence-level bleu+1 yields short translations. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, Bombay, India.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses: An Introduction*. Wiley, New York.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, Edmonton, Cananda.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL)*, Stroudsburg, PA.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, Beijing, China.

Stefan Riezler, Patrick Simianer, and Carolin Haas. 2014. Response-based learning for grounded machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Baltimore, MD.

Patrick Simianer, Stefan Riezler, and Chris Dyer. 2012. Joint feature selection in distributed stochastic learning for large-scale discriminative training in SMT. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, Jeju Island, South Korea.

Jason Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez. 2013. Dirt cheap web-scale parallel text from the Common Crawl. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, Sofia, Bulgaria.

Csaba Szepesvári. 2009. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool.

Andreas Vlachos and Stephen Clark. 2014. A new corpus and imitation learning framework for context-dependent semantic parsing. *Transactions of the Association for Computational Linguistics (TACL)*, 2:547–559.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, Beijing, China.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL)*.

Matthew D. Zeiler. 2012. ADADELTA: An adaptive learning rate method. arXiv:1212.5701 [cs.LG].

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of AAAI*, Portland, OR.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, Edinburgh, Scotland, UK.